Stat Comput (2008) 18: 435–446
DOI 10.1007/s11222-008-9104-9

Differential Evolution Markov Chain with snooker updater and fewer chains

Cajo J.F. ter Braak · Jasper A. Vrugt

Received: 24 July 2008 / Accepted: 24 September 2008 / Published online: 21 October 2008

Abstract Differential Evolution Markov Chain (DE-MC) is an adaptive MCMC algorithm, in which multiple chains are run in parallel. Standard DE-MC requires at least $N = 2d$ chains to be run in parallel, where d is the dimensionality of the posterior. This paper extends DE-MC with a snooker updater and shows by simulation and real examples that DE-MC can work for d up to 50–100 with fewer parallel chains (*e.g.* $N = 3$) by exploiting information from their past by generating jumps from differences of pairs of past states. This approach extends the practical applicability of DE-MC and is shown to be about 5–26 times more efficient than the optimal Normal random walk Metropolis sampler for the 97.5% point of a variable from a 25–50 dimensional Student t_3 distribution. In a nonlinear mixed effects model example the approach outperformed a block-updater geared to the specific features of the model.

Keywords Evolutionary Monte Carlo · Metropolis algorithm · Adaptive Markov chain Monte Carlo · Theophylline kinetics · Adaptive direction sampling · Parallel computing · Differential evolution

1 Introduction

Bayesian statistical analysis combines the data likelihood with a prior distribution using Bayes theorem. A key task is then to summarize the so obtained posterior distribution, for example by the mean, the covariance or percentiles of individual parameters. When this task cannot be carried out by analytical means nor by analytical approximation, simulation methods such as Markov chain Monte Carlo (MCMC) can be used for generating a sample from the posterior distribution. The desired summary of the posterior distribution is then obtained from the sample. The posterior distribution, also referred to as the target, is typically high dimensional.

Ter Braak (2006) proposed a simple adaptive random walk Metropolis algorithm called Differential Evolution Markov Chain (DE-MC). DE-MC is local Differential Evolution (Storn and Price 1997; Price et al. 2005) with an added Metropolis step. DE-MC solves an important practical problem in random walk Metropolis, namely that of choosing an appropriate scale and orientation for the jumping distribution. Earlier approaches such as (parallel) adaptive direction sampling (Gilks et al. 1994; Roberts and Gilks 1994; Gilks and Roberts 1996) solved the orientation problem but not the scale problem.

In DE-MC N different chains are run in parallel and the jumps for each chain are derived from the remaining $N - 1$ chains. Each jump is simply generated as follows (Fig. 1a). The difference of two vectors of randomly chosen chains is multiplied by a factor γ, a random vector is drawn from a narrow symmetrical distribution and the sum of the scaled difference and the random vector is added to the vector of the current chain. This jump is reversible as Fig. 1b illustrates. The difference vector contains the required information on scale and orientation. By accepting each jump with the Metropolis ratio, a Markov chain is obtained, the stationary distribution of which is the posterior distribution. The proof of this uses the fact that DE-MC is essentially a Metropolis-within-Gibbs algorithm (see Appendix). Ter Braak (2006) showed that a sensible choice

C.J.F. ter Braak (✉)
Biometris, Wageningen University and Research Centre,
Box 100, 6700 AC, Wageningen, The Netherlands
e-mail: Cajo.terbraak@wur.nl

J.A. Vrugt
Center for NonLinear Studies (CNLS), Los Alamos National Laboratory, Los Alamos, NM 87545, USA

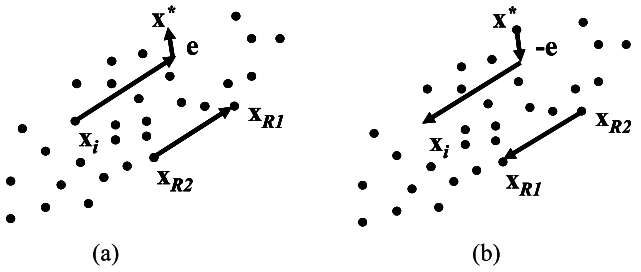

Fig. 1 Parallel direction update in DE-MC in two dimensions with 30 parallel chains, each represented by a point ($d = 2$, $N = 30$). (**a**) For updating the ith chain, which is in state $\mathbf{x}_i$, the proposed state is $\mathbf{x}^*$, generated from $\mathbf{x}_i$, the difference of the states of two other chains ($\mathbf{x}_{R1}$ and $\mathbf{x}_{R2}$) and a random vector $\mathbf{e}$ by (1) with $\gamma = 2.4/(2 \times 2)^{1/2} = 1.2$. (**b**) The reverse jump from $\mathbf{x}^*$ to $\mathbf{x}_i$ is obtained by reversing $\mathbf{e}$ and the order of the two other states

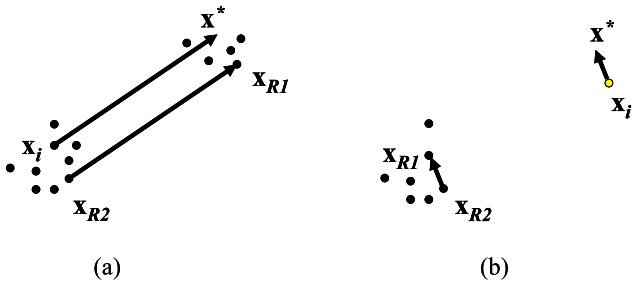

Fig. 2 Parallel direction update in DE-MC in two dimensions. (**a**) DE-MC can jump between modes with $\gamma = 1.0$ and (**b**) an outlier chain ($\mathbf{x}_i$) may need considerable time to reach the modal region because the differences within this region are small. In (**a**) and (**b**) the random term $\mathbf{e}$ is neglected as it is small compared to the jumps

of γ is $2.38/\sqrt{(2d)}$, with d the number of parameters of the posterior. This choice is motivated by comparison with random walk Metropolis with a normal jumping distribution (RWMN) (Roberts and Rosenthal 2001). This choice of γ gives, at least for Gaussian and Student target distributions, DE-MC acceptance probabilities close to 0.44 for $d = 1$, 0.28 for $d = 5$ and 0.23 for large d (see Sect. 7.84 of Robert and Casella (2004) for a cautionary note on these references acceptance rates). After a burn-in period, the states of the chains are independent so that convergence of a DE-MC run can be monitored with the $\hat{R}$-statistic of Gelman et al. (2004). DE-MC shares this useful feature with other population MCMC samplers (Mengersen and Robert 2003).

DE-MC can be effective to explore multimodal densities. With an occasional choice of $\gamma \approx 1$, a chain can jump between two disconnected modes (Fig. 2a). If a mode is represented by at least a single chain, a second chain can be moved to it in accordance with the posterior mass of the mode. This simple strategy of DE-MC balances exploration and exploitation of the space.

For DE-MC to work well the number of chains N must be larger than d. Our previous work has shown that $N = 2d$ or $3d$ worked fine for simple unimodal posteriors for $d < 50$, say, but that $N = 10d$ to $20d$ was required for more complicated posteriors (ter Braak 2006). A large N has a disadvantage though. When initialized from a wide prior, each chain must travel to the high density region of the posterior. Although jumps can initially be larger than in RWMN, the time for all N chains to converge is typically a factor N larger than for a single chain. For slowly converging adaptive chains the performance could even be worse.

There are two other reasons why using a smaller N might be advantageous. First, if the posterior is unimodal and all but one chain have converged to the modal region, it might still take considerable time to also move this outlier chain to the mode, irrespective of the value of γ (Fig. 2b). Consequently, standard DE-MC has potentially an outlier problem. Empirically outlier chains occur more often with large N which is necessary for large d. Second, in a multi-processor environment, chains could run on individual computational nodes (processors). The lower the number of nodes required, the greater the practical applicability of DE-MC for computationally demanding problems. It would then also be advantageous that the proposal of the ith chain would not require the updated states of the chains $1, \ldots, (i-1)$, as they do in Metropolis-within-Gibbs and thus in standard DE-MC. There is therefore sufficient scope to further increase the efficiency and implementation of DE-MC.

One device that allows for the use of smaller N is to decrease the number of parameters that is simultaneously altered in each jump. Rather than performing a full-dimensional update, one can update blocks of parameters in turn. With blocks of one parameter, each parameter is updated in turn as in Gibbs sampling. More generally, the parameters to be updated jointly can be selected randomly with some probability CR, the crossover rate (Price et al. 2005; Vrugt et al. 2008a, 2008b). Preferably, highly correlated parameters should be updated jointly; so better probabilistic schemes can be devised. An extreme case is to fix the blocks of parameters in advance as illustrated in ter Braak (2006) for a nonlinear mixed-effects model. The model in question had $d = 43$ and was sampled with blocks of size one to three using $N = 9$. Some computational tricks and special features of the model were used to let the method outperform standard DE-MC using $2d = 86$ chains with full-space updates.

This paper explores another way to decrease N, namely by sampling the difference vectors in the DE-MC jump from past states, which turns the method into an adaptive Metropolis sampler (Haario et al. 2001; Roberts and Rosenthal 2007, 2008).

It is always of interest to have a larger variety of efficient update schemes. In analogy with adaptive direction sampling (Gilks et al. 1994), we present a snooker update for DE-MC. Gibbs sampling usually samples along each coordinate axis in turn (each representing a parameter). A snooker update also samples along one axis at a

time, but this axis does not need to run parallel to the coordinate axes. The snooker axis typically runs through the states of two different chains. Gibbs sampling along this axis is often not feasible, and therefore one needs to resort to a Metropolis-Hastings update. The DE-MC snooker update presented herein is such an update, but with an adaptive step size.

The effectiveness of the proposed methods is demonstrated using two known synthetic target distributions (Student t_{60} and t_3) and two Bayesian data analysis examples.

2 Theory

2.1 Standard DE-MC

Let the states of the N chains be denoted by the d-dimensional parameter vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ where d is the number of parameters. Together the chains form a population that evolves through generation time. In a Bayesian analysis the initial population could be drawn from the prior distribution of the parameters. For each chain i in turn ($i = 1, \ldots, N$), a jump is simply generated as (Fig. 1a)

$$\mathbf{x}^* = \mathbf{x}_i + \gamma(\mathbf{x}_{R1} - \mathbf{x}_{R2}) + \mathbf{e}, \tag{1}$$

where $\mathbf{x}_{R1}$ and $\mathbf{x}_{R2}$ are randomly selected without replacement from the population $\mathbf{X}_{-i}$ (the population without $\mathbf{x}_i$), γ is a user-defined scalar, and $\mathbf{e}$ is drawn from a symmetric distribution with a small variance compared to that of the posterior, but with unbounded support, *e.g.* $\mathbf{e} \sim \mathrm{N}(0, b)^d$ with b small. As each jump is as likely as the reverse jump, given the current state of the remaining chains (Fig. 1b), conditional detailed balance with respect to $\pi(.)$ is obtained by accepting each jump with the Metropolis ratio $\min(1, r)$ where $r = \pi(\mathbf{x}^*)/\pi(\mathbf{x}_i)$. All N chains together form a Metropolis-within-Gibbs sampler on an $N \times d$-dimensional space. Conditional detailed balance for each chain is then sufficient to show that the resulting joint Markov chain has a stationary distribution with a density that factorizes in to N terms that are all equal to the density of the posterior distribution (Mengersen and Robert 2003). Because of the unbounded support of $\mathbf{e}$ in (1), the joint chain is ergodic, each of the N chains converges to the posterior distribution, and at any time after convergence the N chains are independent. The proof is given in the Appendix.

Upon convergence, the averages across the population at each generation, converge for large N to the expectation and covariance of the posterior distribution, *i.e.*

$$\mathrm{ave}(\mathbf{x}_i) \to \boldsymbol{\mu} \quad \text{and} \quad \mathrm{ave}[(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}] \to 2\boldsymbol{\Sigma}$$
$$\text{for } N \to \infty$$

with ave the average across the (pairs of) chains. For large N and small b, the proposal in (1) thus looks like $\mathbf{x}^* = \mathbf{x}_i + \gamma\boldsymbol{\varepsilon}$ with $\mathrm{E}(\boldsymbol{\varepsilon}) = \mathbf{0}$ and $\mathrm{cov}(\boldsymbol{\varepsilon}) = 2\boldsymbol{\Sigma}$, the covariance matrix of the target. In particular, if $\pi(.)$ is multivariate normal, then $\gamma\boldsymbol{\varepsilon} \sim \mathrm{N}(0, 2\gamma^2\boldsymbol{\Sigma})$ so that DE-MC is expected to behave like RWMN. From the guidelines for c in RWMN (Roberts and Rosenthal 2001) a sensible choice of γ is $2.38/\sqrt{(2d)}$. This choice of γ gives, for Gaussian and Student target distributions, DE-MC acceptance probabilities close to 0.44 for $d = 1$, 0.28 for $d = 5$ and 0.23 for large d. If the initial population is drawn from the prior, DE-MC translates the 'prior population' to the 'posterior population'.

The jump in (1) is almost parallel to the line $\mathbf{x}_{R1} - \mathbf{x}_{R2}$ and is therefore called the parallel direction update. Parallel adaptive direction sampling (Gilks et al. 1994; Roberts and Gilks 1994) uses Gibbs sampling along this direction (when feasible), whereas (1) uses a local move. Without the random term $\mathbf{e}$ in (1), all updates lie in a space of dimension $\min(d, N - 1)$. To efficiently sample the full space, N must therefore be larger than d.

2.2 DE-MC$_Z$: sampling the difference vectors from the past

One way to circumvent the requirement that $N > d$ is to sample the difference vectors in the update of (1) from past states of the chains. This turns DE-MC into an adaptive MCMC method in the sense of Haario et al. (2001) and Roberts and Rosenthal (2007). To ensure that the chain converges to the posterior distribution, the adaptation should decrease in time (Roberts and Rosenthal 2007, 2008). This can be achieved by sampling the difference vectors (which form the adaptive part of our proposal) uniformly at random without replacement from a possibly thinned, version of the entire past. Thinning has important advantages, as it reduces storage requirements.

We implement this idea as follows. Let $\mathbf{X}$ denote an $N \times d$ matrix that stores the locations of the chains at the current generation, and $\mathbf{Z}$ the matrix that contains the current and past states of the chains. DE-MC only contains two algorithmic parameters, N and γ. Compared to DE-MC, DE-MC$_Z$ contains two additional variables; M_0, the initial number of rows of $\mathbf{Z}$, and K which defines the thinning rate. As default choice, we set $N = 3$, $\gamma = 2.38/\sqrt{(2d)}$, $M_0 = 10d$ and $K = 10$.

Algorithm DE-MC$_Z$

1. Initialize the $M_0 \times d$ matrix $\mathbf{Z}$, for example by sampling from the prior distribution with $M_0 > \max(d, N)$, copy the first N rows of $\mathbf{Z}$ to $\mathbf{X}$ and set $M \leftarrow M_0$.
2. K times update population $\mathbf{X}$.
3. Append the current rows of $\mathbf{X}$ to $\mathbf{Z}$, so that $M \leftarrow M + N$.
4. If $\mathbf{X}$ has converged or the total number of updates of $\mathbf{X}$ is greater than G, go to 5, else go to step 2.

5. Summarize the samples stored in $\mathbf{Z}$ after discarding the initial and burn-in samples.

In this algorithm, each update of $\mathbf{X}$ forms one generation cycle and sequentially updates $\mathbf{x}_1, \ldots, \mathbf{x}_N$ as follows.

Algorithm for updating population X

For $i = 1, \ldots, N$ do

1. Sample uniformly at random without replacement two numbers R_1 and R_2 from the numbers $1, 2, \ldots, M$.
2. Calculate the proposal $\mathbf{x}^*$ (Figs. 1 and 2)

$$\mathbf{x}^* \leftarrow \mathbf{x}_i + \gamma(\mathbf{z}_{R1} - \mathbf{z}_{R2}) + \mathbf{e} \tag{2}$$

   where $\mathbf{z}_{R1}$ and $\mathbf{z}_{R2}$ are rows R_1 and R_2 of $\mathbf{Z}$ and γ and $\mathbf{e}$ are as in (1).
3. Calculate the Metropolis ratio $r = \pi(\mathbf{x}^*)/\pi(\mathbf{x}_i)$.
4. Accept the proposal, *i.e.* $\mathbf{x}_i \leftarrow \mathbf{x}^*$ with probability $\min(1, r)$, otherwise leave $\mathbf{x}_i$ unchanged.

The matrix $\mathbf{Z}$ grows through time. Appending N rows changes $\mathbf{Z}$ by an order of $N/M = K/t$, which decreases in generation time t, as required in adaptive MCMC. Changes in the proposal distribution (and therefore in the transition kernel) from one batch of iterations to the next therefore decrease to zero as the length of the thinned past increases without bound. Because of this diminishing adaptation (Roberts and Rosenthal 2007), we conjecture that DE-MC$_Z$ is ergodic and converges to a stationary distribution with pdf $\pi(.)^N$. Note that the chain may converge to another than the intended distribution if only the recent past is used to generate the jumps (see Haario et al. 2001 for an example)—even if the vectors generating the difference are never sampled from the past of the current chain.

The following remarks are in order. The algorithm is similar in spirit to the adaptive Metropolis sampler (AM) of Haario et al. (2001). AM is a single chain RWMN sampler using a covariance matrix that is based on all past samples thinned at rate K. The full past is required to guarantee ergodicity of the chain (Haario et al. 2001). DE-MC$_Z$ also needs to use the full past. A similar idea was explored in DE (Babu and Angira 2006).

Because the asymptotic joint pdf of the N chains factorizes to $\pi(\mathbf{x}_1) \times \cdots \times \pi(\mathbf{x}_N)$, the states $\mathbf{x}_1, \ldots, \mathbf{x}_N$ of the individual chains are independent at any generation after DE-MC has become independent of its initialization. This feature of population MCMC samplers (Mengersen and Robert 2003) is important for monitoring the convergence of a DE-MC and DE-MC$_Z$ run with the $\hat{R}$-statistic of Gelman et al. (2004). This statistic compares for each scalar parameter of interest the between- and within-variance of the chains. Because of the asymptotic independence, the between-chain variance and $\hat{R}$ can be estimated consistently from a single DE-MC$_{(Z)}$ run. Gelman et al. (2004) consider $\hat{R}$ below 1.2 acceptable.

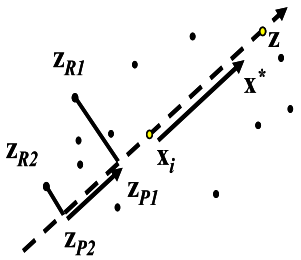

Fig. 3 The DE snooker update, which generates a proposal along the line through $\mathbf{x}_i$ and state $\mathbf{z}$ of another chain. The proposal point $\mathbf{x}^*$ is generated by randomly selecting two other chains ($\mathbf{z}_{R1}$ and $\mathbf{z}_{R2}$), by projecting them orthogonally on to the line ($\mathbf{z}_{P1}$ and $\mathbf{z}_{P2}$) and adding a multiple (1.7) of the difference between the projection points $\mathbf{z}_{P1}$ and $\mathbf{z}_{P2}$ to $\mathbf{x}_i$. DE-MC$_S$ and DE-MC$_{ZS}$ use 10% snooker updates and 90% parallel direction updates

2.3 Snooker update

In this section, a DE-MC snooker update is proposed. The standard snooker update has been presented in Gilks et al. (1994), Roberts and Gilks (1994), Liang and Wong (2001) and works as follows

To update $\mathbf{x}_i$

1. Select another chain, which is in state $\mathbf{z}$, say.
2. Gibbs sample along the line $\mathbf{x}_i - \mathbf{z}$ from the density $g(\mathbf{x})$ on that line; with $\mathbf{x}$ a point on the line (Liang and Wong 2001)

$$g(\mathbf{x}) \propto \pi(\mathbf{x}) \|\mathbf{x} - \mathbf{z}\|^{d-1}.$$

Often it is not directly feasible to perform step 2, and one has to resort to Griddy-Gibbs sampling, adaptive rejection sampling (ARS) or a Metropolis-Hastings step (Gilks et al. 1994; Liang and Wong 2001). Griddy-Gibbs or ARS would require multiple evaluations of the full posterior and is therefore likely not very efficient. A random walk Metropolis-Hastings step gives as always a step-size problem that a multi-chain method can solve automatically. This is the key to the snooker update in DE-MC, which we describe now. This snooker update is a Metropolis step with adaptive step size.

Algorithm of DE Snooker update (Fig. 3)

To update $\mathbf{x}_i$

1. Select another chain, which is in state $\mathbf{z}$, say.
2. Sample along the line $\mathbf{x}_i - \mathbf{z}$ from the density on that line as follows (Fig. 3).
   a. Select two other random chains, R_1 and R_2 that are in states $\mathbf{z}_{R1}$ and $\mathbf{z}_{R2}$ respectively.
   b. Project $\mathbf{z}_{R1}$ and $\mathbf{z}_{R2}$ orthogonally on to the line $\mathbf{x}_i - \mathbf{z}$ yielding $\mathbf{z}_{P1}$ and $\mathbf{z}_{P2}$.
   c. Propose,

$$\mathbf{x}^* \leftarrow \mathbf{x}_i + \gamma_s(\mathbf{z}_{P1} - \mathbf{z}_{P2}). \tag{3}$$

d. Calculate the Metropolis ratio

$$r = \frac{\pi(\mathbf{x}^*)\|\mathbf{x}^* - \mathbf{z}\|^{d-1}}{\pi(\mathbf{x})\|\mathbf{x}_i - \mathbf{z}\|^{d-1}}. \quad (4)$$

e. Accept the proposal with probability $\min(1, r)$, otherwise remain at $\mathbf{x}_i$.

Remarks (1) The choice of γ_s is similar to that in DE-MC, except that it uses $d = 1$ for all d, *i.e.* $\gamma_s = 2.38/\sqrt{2} \approx 1.7$. The reason for this is that the projection step already reduces the variance of the difference and thus takes care of the dimensionality. This choice gives approximately the same acceptance rate for Normal and Student posteriors as in a well-scaled Metropolis-Hastings algorithm. At the expense of some inefficiency for Normal and Student posteriors, γ_s can be taken at random, for example, uniform in a unit interval centred around 1.7, *e.g.* $\gamma_s \sim \mathrm{U}[1.2, 2.2]$. This is our default setting. Since the one-dimensional slice could have any distribution, there is not necessarily a loss in efficiency.

(2) In step 1 of both algorithms, we chose the state of another chain to compute the direction. Alternatively, $\mathbf{z}$ can be replaced by any other state (Liang and Wong 2001). Possibilities are: the average of states of a number of other, possibly best, chains or simply $\mathbf{z} + \mathbf{e}$ with $\mathbf{e}$ as in (1).

(3) Equation (4) nicely shows that a reversible chain in $d > 1$ cannot jump directly from $\mathbf{x}_i$ to $\mathbf{z}$, since with such a proposal $\|\mathbf{x}^* - \mathbf{z}\| = 0$ and, hence, $r = 0$. For $d = 1$, the snooker update reduces to (2) with $\mathbf{e}$ equal to zero.

(4) An interesting extension of the snooker update is to choose the sign and value of γ_s depending on the values of the posterior at $\mathbf{x}_i$ and $\mathbf{z}$. If $\pi(\mathbf{z}) > \pi(\mathbf{x}_i)$, then one would like to jump with higher probability than 0.5 towards $\mathbf{z}$ and if $\pi(\mathbf{z}) < \pi(\mathbf{x}_i)$, then one would like to jump with higher probability than 0.5 away from $\mathbf{z}$. Of course, we need a Metropolis-Hastings correction for this bias in the jump direction. One way of implementing this idea is to jump towards $\mathbf{z}$ with probability

$$\Phi\left(c^*\left(\log(\pi(\mathbf{z})) - \log(\pi(\mathbf{x}_i))\right)\right)$$

with $\Phi(.)$ the cumulative standard normal distribution and c^* an adjustable parameter. This snooker DE update is constructed in the spirit of more 'intelligent' algorithms such as the Metropolis-adjusted Langevin algorithm (Roberts and Rosenthal 2001). Limited experimentation with this idea did not show any real advantage over the simple DE Snooker update.

(5) In this paper, the DE snooker update is always mixed with the parallel direction update so as to diversify the jumping possibilities. After some experimentation we chose the updates in a mix of 10% snooker updates and 90% parallel direction updates. We denote the resulting sampler with DE-MC$_\mathrm{S}$ and, when used with differences of past samples, by DE-MC$_\mathrm{ZS}$ and both by DE-MC$_\mathrm{(Z)S}$.

3 Tests with known distributions

DE-MC and DE-MC$_\mathrm{Z}$ with and without snooker update were applied to multivariate Student distributions with sixty and with three degrees of freedom, both distributions centred at the zero vector. The covariance matrix was set such that the variance of the jth variable was equal to j and all pairwise correlations were 0.5. These distributions were chosen to reflect the possibly widely differing scales of unknown parameters in many applications. These samplers were also compared to the optimal RWMN sampler and a RWMN sampler (MH-est) in which the covariance matrix of the proposal was estimated from burn-in draws and the scaling factor was set such that the acceptance rate was about 0.24. The initial covariance matrix was the identity matrix multiplied with the average true variance of the variables.

In all simulations and analyses, the details of the tested samplers were as follows. To allow for occasionally large jumps with the parallel direction updates of (1) and (2), $\gamma = 1$ with probability 0.1 (Fig. 2) and $\gamma = 2.38/\sqrt{(2d)}$ otherwise (Fig. 1). We used $\mathrm{var}(\mathbf{e}) = b = 10^{-4}$. When the snooker update (3) was included, it was applied to 10% of the updates with $\gamma_s \sim \mathrm{U}[1.7, 2.2]$. DE-MC and DE-MC$_\mathrm{S}$ were run with $N = 2d$, DE-MC$_\mathrm{Z(S)}$ with $N = 3$, $K = 10$ and optimal and estimated RMWN with $N = 1$. The initial population was drawn from the d-dimensional uniform distribution $\mathrm{U}[-5, 15]^d$, reflecting a lack of prior knowledge about the mean and variance of the posterior.

The efficiency of each sampler for a given statistic is defined with respect to the optimal RWMN as $100 \times \mathrm{MSE}_\mathrm{optRWMN}/\mathrm{MSE}_\mathrm{sampler}$, where MSE is the mean squared error in the statistic. The statistics we used were the empirical 2.5, 50 and 97.5-percentiles which, for a d-dimensional distribution, were determined from the sample for the first and dth variable. The squared error divided by the true variance of the variable did not differ much between these variables and therefore their mean was used in the calculation of the MSE. Because the theoretical MSEs for the 2.5 and 97.5 percentiles are equal, their estimated MSEs were averaged and their average was used to calculate the efficiency under the heading P2.5. It is thus a pooled efficiency for the 2.5 and 97.5 percentiles.

Each efficiency estimate is based on 100 runs, each consisting of 10^6 draws of each sampler after a burn-in of 10^5 draws, with draws counting the number of proposal evaluations, each one requiring one evaluation of $\pi(.)$. With $K = 10$, the final number of rows of $\mathbf{Z}$ in DE-MC$_\mathrm{Z(S)}$ is thus $10d + 1.1 \times 10^5$ for any value of N.

We would like to stress that our comparison of DE-MC and DE-MC$_\mathrm{Z(S)}$ with the RWMN samplers is fair in the sense that each run required the same number of evaluations of the target function $\pi(.)$ and the respective statistics (parameters) were calculated using the same number

Table 1 Efficiency (in percentages) of DE-MC variants with respect to Random Walk Metropolis with optimal Normal jumping distribution for the median (P50) and 2.5% percentile (P2.5) of d-dimensional Student t_{60} and t_3 distributions. (DE-MC: default, parallel direction updates only; subscript S: 10% snooker updates, 90% parallel direction updates; subscript Z: updates use sampling difference vectors from past)

	N	Student t_{60}						Student t_3			
		$d=25$		$d=50$		$d=100$		$d=25$		$d=50$	
		P50	P2.5	P50	P2.5	P50	P2.5	P50	P2.5	P50	P2.5
MH-est.	1	69	82	94	61	79	38	33	77	15	143
DE-MC	$2d$	68	71	77	73	10	5	67	10	27	3
DE-MC$_S$	$2d$	63	77	88	86	9	31	76	439	108	1016
DE-MC$_Z$	3	85	94	108	85	13	17	128	125	27	114
DE-MC$_{ZS}$	3	88	99	106	103	70	79	117	506	131	2668

Note. The estimated MSEs per draw of RWMN were, in column order, 85, 285, 203, 552, 384, 951, 65, 13916, 166 and 102010. P2.5 is a pooled efficiency for the 2.5 and 97.5 percentiles

of draws. In real applications the function evaluation costs typically dominate CPU time, in which case all runs would require about the same amount of CPU time. To be specific, in Sects. 3.1 and 3.2 each RMWN run consisted of 1.1×10^6 iterations of a single chain of which 10^5 iterations were discarded as burn-in, whereas DE-MC or DE-MC$_{Z(S)}$ runs with N parallel chains consisted of $1.1 \times 10^6/N$ generations (updates of $\mathbf{X}$) of which $10^5/N$ generations were discarded as burn-in. For $d = 50$, for example, DE-MC and DE-MC$_S$ were run with $N = 100$. Each such run consisted of 11,000 generations of which 1,000 were discarded as burn-in. RMWN could thus benefit from $N = 100$ times more burn-in iterations than each of the member chains of DE-MC.

3.1 Multivariate Student distribution with 60 degrees of freedom

Table 1 shows the efficiency of the samplers with respect to RWMN with the optimal jumping distribution (with $c = 2.38/\sqrt{d}$ and $\mathbf{\Sigma}$ set to the true covariance of the distribution) as obtained from a simulation study for $d = 25, 50$ and 100.

DE-MC$_{ZS}$ (with sampling difference vectors from the past and with 10% snooker updates) showed the best performance with efficiencies ranging between 70% and 106%, followed by MH-est that exhibits the second highest efficiency ranging between 38% and 94%. The efficiencies of standard DE-MC were all about 70%, except for $d = 100$, which shows a much lower efficiency. This loss was due to bias, despite the fact that each individual run had converged as judged on the basis of the $\hat{R}$ statistics (all $\hat{R} < 1.2$). If the efficiency were based on the estimated variance or if the length of the burn-in period would have been doubled, then the efficiency of DE-MC would still be about 70% as shown in ter Braak (2006). DE-MC$_S$ tended to perform slightly better than DE-MC.

3.2 Multivariate Student distribution with three degrees of freedom

For a Student t_3 distribution, the samplers with snooker update performed much better than the other samplers, in particular for the 2.5% percentile, with a 4–27 times improvement over the optimal RWMN. Sampling difference vectors from the past increased the efficiency, as DE-MC$_Z$ and DE-MC$_{ZS}$ had higher efficiencies than both DE-MC and DE-MC$_S$, respectively. MH-est and DE-MC did poorly. Ter Braak (2006) reported high efficiencies for DE-MC for $d = 50$ but these results were obtained using a better initial population.

We also carried out runs for $d = 100$ with 10^6 draws. The 50% and 2.5% percentiles of the Student t_3 with unit variance 1, are 0 and 1.84, respectively. On this scale, the Root Mean Squared Error (RMSE) using the optimal RWMN were 0.015 and 0.61 for the 50% and 2.5% percentiles, respectively, whereas DE-MC$_{ZS}$ resulted in 0.035 and 0.09, respectively. In this case, DE-MC$_{ZS}$ is about 40 times more efficient for the 2.5% percentile.

Acceptance rates in all these runs varied between 0.21 and 0.24. In optimal RWMN and MH-est, c was scaled to this acceptance rate, whereas the DE-MC variants gave this rate automatically using $\gamma = 2.38/\sqrt{(2d)}$.

3.3 Effect of N and number of draws on efficiency

Whereas Table 1 varied the dimension and the number of degrees of freedom for a fixed number of draws and parallel chains ($N = 3$) in DE-MC$_{(Z)(S)}$, Table 2 varies the number of draws and the number of parallel chains (N) for a fixed dimension $d (d = 10)$ and degrees of freedom (3). The target distribution is thus a 10-dimensional t_3 distribution and the initial population was as before and thus overdispersed and far removed from the target.

Table 2 Mean squared error (per 1000 draws) in the 2.5% percentile of a 10-dimensional Student t_3 distribution for the samplers of Table 1 in relation to the number of draws and the number of parallel chains (N)

Sampler	N	Number of draws			
		5×10^3	10^4	2×10^4	10^6
RMWN-opt	1	47.2	49.3	45.4	2.9
MH-est	1	21.1	23.2	35.5	4.1
DE-MC	20	162.1	244.3	314.0	3.8
DE-MC$_S$	20	128.2	128.4	228.3	1.3
DE-MC$_{ZS}$	1	3.0	1.5	1.2	1.0
DE-MC$_{ZS}$	2	3.5	1.5	1.2	1.2
DE-MC$_{ZS}$	4	5.4	2.3	1.3	1.1
DE-MC$_{ZS}$	8	16.1	6.5	1.9	1.1
DE-MC$_{ZS}$	16	47.7	27.5	11.2	1.0

Table 2 shows the mean squared error per draw (MSEpD) of the samplers for the 2.5% percentile (defined as in Table 1) using 1,000 independent runs with 5×10^3, 10^4, and 2×10^4 draws and an additional 200 runs with 10^6 draws. The first 10% of the draws of each run was discarded and used as burn-in. Table 2 does not include DE-MC$_Z$ as DE-MC$_{ZS}$ consistently performed somewhat better.

Table 2 shows that DE-MC$_{ZS}$ with $N \leq 8$ had much lower MSEpD than the RWMN and DE-MC$_{(S)}$ samplers. For 10^4 draws, DE-MC$_{ZS}$ with $N = 1 - 2$ is about 30 times more efficient than the optimal RWMN, whereas for 10^6 draw it is approximately 2.5 times more efficient. Notice that, with N between 1 and 4, 10^6 draws with either implementation of RWMN yields higher MSEpDs than 10^4 draws with DE-MC$_{ZS}$. Table 2 also shows that, for a small number of draws, DE-MC (without sampling difference vectors from the past) can be very inefficient compared to RWMN.

Inflating N increases MSEpD in DE-MC$_{ZS}$ when using 5×10^3 draws, but this effect of N decreases with increasing numbers of draws, eventually disappearing after 10^6 draws. The reason is that it takes time for the chain(s) to converge from a distant point towards the target, but after convergence the efficiency is largely independent of N. Judged by this criterion, DE-MC$_{ZS}$ with $N = 1$–2 has converged within 2×10^4 draws. Indeed, with $N = 2$ the Gelman's $\hat{R}$ statistic never exceeded 1.2 with 2×10^4 draws, but did so in 7% of the runs with 10^4 draws.

The conclusion we draw from Table 2, is that N should be chosen small in DE-MC$_{ZS}$. In further analyses, we choose $N = 3$ as it allows for a better assessment of convergence with the $\hat{R}$ statistic of Gelman et al. (2004).

4 Bayesian examples

4.1 One-way random-effects model

Ter Braak (2006) presented a DE-MC analysis of a one-way random-effects model for four groups yielding seven parameters in total using an example from Liu and Hodges (2003). The example is more difficult than one might think because of a discrepancy between the prior and the data likelihood. The DE-MC analysis showed a small bias in $\log(\xi)$ where ξ denotes the variance components ratio, when the number of simultaneous chains was increased from 14 to 21 to 70 using 10^6 draws. Here we analyze the same data using the DE-MC variants presented herein using 10^4 draws of which the first 20% was discarded and used as burn-in. With this number of draws, standard DE-MC with $N = 14$ converged in 86 of the 100 runs, whereas the DE-MC variants converged in all cases (Table 3), of which DE-MC$_{ZS}$ showed the closest mean and smallest standard deviation of the percentiles of $\log(\xi)$ in Table 3. Acceptance rates in all these runs varied between 0.20 and 0.26.

Table 3 Number of runs with $\hat{R} < 1.2$ and mean (standard deviation) of percentiles in 100 runs of the posterior of $\log(\xi)$ of a one-way random-effects model with $d = 7$

Sampler	N	#($\hat{R} < 1.2$)	$\log(\xi)$		
			P2.5	P50	P97.5
True			−0.94	0.98	4.00
DE-MC	14	86	−0.95	1.01	4.05
			(0.14)	(0.10)	(0.33)
DE-MC$_S$	14	100	−0.88	1.08	4.01
			(0.13)	(0.11)	(0.25)
DE-MC$_Z$*	3	100	−0.87	1.06	3.98
			(0.13)	(0.16)	(0.22)
DE-MC$_{ZS}$*	3	100	−0.91	1.01	4.00
			(0.13)	(0.09)	(0.22)

Note. Each run consisted of 10^4 draws of which 20% burn-in. * with $K = 1$

4.2 Nonlinear mixed-effects model

This subsection illustrates the advantage of using fewer chains in DE-MC for a nonlinear mixed-effects model us-

Table 4 Root mean squared error of percentiles for DE-MC ($N = 86$, 5000 generations), DE-MC$_{ZS}$ ($N = 3$, 143, 333 generations, $M_0 = 430$, $K = 3$), and Block DE-MC ($N = 9$ and 5000 generations with two inner iterations), based on 41, 100 and 75 runs out of 100 runs with 20% burn-in and maximum $\hat{R} < 1.2$ and requiring *ca.* 25 seconds per simulation each on a 3.2 GHz Pentium 4, respectively

	DE-MC $N = 86(41/100)$			DE-MC$_{ZS}$ $N = 3(100/100)$			Block $N = 9(75/100)$		
	P2.5	P50	P97.5	P2.5	P50	P97.5	P2.5	P50	P97.5
lKe	0.010	0.002	0.007	0.004	0.002	0.003	0.009	0.004	0.009
lKa	0.021	0.008	0.028	0.025	0.011	0.036	0.069	0.025	0.076
lCl	0.007	0.002	0.007	0.007	0.003	0.006	0.016	0.006	0.017
$\log(\tau_e^2)$	3.231	0.496	0.228	2.965	0.246	0.070	1.202	0.231	0.108
$\log(\tau_a^2)$	0.040	0.017	0.057	0.021	0.021	0.049	0.027	0.026	0.075
$\log(\tau_c^2)$	0.034	0.043	0.089	0.029	0.023	0.039	0.031	0.027	0.063
$\log(\sigma^2)$	0.006	0.004	0.022	0.007	0.006	0.009	0.005	0.003	0.006

ing the Theophylline data presented in Pinheiro and Bates (2000, p. 444). The data consist of the oral doses of the anti-asthmatic drug Theophylline administered to twelve patients and the serum concentrations of Theophylline in these patients at 11 time points over 25 hours after the oral intake. The pharmacokinetics of this drug is modeled by the first-order open-compartment model

$$\mu_{it} = \frac{D_i k_{ei} k_{ai}}{c_i (k_{ai} - k_{ei})} \left[\exp(-k_{ei} t) - \exp(-k_{ai} t)\right]$$

where μ_{it} is the expected concentration of the ith patient at time t, D_i is the dose of theophylline administered to the ith patient and k_{ei}, k_{ai} and c_i are unknown patient-specific parameters representing the elimination rate, absorption rate and clearance, respectively. As in ter Braak (2006), analysis 2 in Pinheiro and Bates (2000, pp. 364–365) was mimicked by using the normal likelihood $y_{it} \sim \mathrm{N}(\mu_{it}, \sigma^2)$, the independent normal priors $\log(k_{ei}) \sim \mathrm{N}(lKe, \tau_e^2)$, $\log(k_{ai}) \sim \mathrm{N}(lKa, \tau_a^2)$ and $\log(c_i) \sim \mathrm{N}(lCl, \tau_c^2)$ and improper uniform priors for *lKe*, *lKa*, *lCl* and $\log \sigma^2$. The priors for the τ-parameters were chosen to be improper uniforms on the τ-scale, *i.e.* $p(\log(\tau_x^2)) \propto \tau_x$, for $x = e, a, c$. The total number of parameters in the posterior density is $3 + 3 + 1 + 12 \times 3 = 43$ of which 36 are patient-specific ones. The log-posterior was programmed and the initial population for DE-MC was drawn in a similar way as done in ter Braak (2006).

Here we compare DE-MC$_{ZS}$ with $N = 3$, $M_0 = 10d$ and $K = 3$ with standard DE-MC with $N = 2d = 86$ and block DE-MC with $N = 9$ as specified in ter Braak (2006). The runs in ter Braak (2006) used 4.3 million draws in DE-MC. Despite this large number, standard DE-MC with $N = 86$ converged only in 71 of the 100 runs. Separate runs with WinBUGS 1.4 (Spiegelhalter et al. 2003) lasted about twice as long, but did not perform well either (ter Braak 2006). In contrast, DE-MC$_{ZS}$ consistently converged in all 100 different sampling runs, whereas block DE-MC in almost all. The advantage of DE-MC$_{ZS}$ can be shown even more convincingly by reducing the number of draws by a factor of ten. Again, DE-MC$_{ZS}$ converged in all 100 runs, whereas standard DE-MC and block DE-MC converged in only 41 and 75 of the 100 runs, respectively. Table 4 compares the samplers in terms of the RMSE of the percentiles of parameters, the true values being based on a very long WinBUGS 1.4 run. As expected, the RMSEs are typically about a factor $\sqrt{10}$ higher than their counterparts presented in ter Braak (2006), which took ten times longer to compute. The RMSEs of DE-MC$_{ZS}$ are similar to or lower than those of DE-MC. Block DE-MC performs best on σ^2 and τ_e^2 (which is very close to 0 and thus ill-determined on a log-scale), whereas the full space methods DE-MC and DE-MC$_{ZS}$ do better for the location parameters *lKe*, *lKa* and *lCl*. Note that block DE-MC required several additional tricks to speed up the computation by using specific properties of the nonlinear effects model, whereas DE-MC$_{ZS}$ did not require additional tuning. Based on these findings, DE-MC$_{ZS}$ appears to be the most efficient and robust sampling method. DE-MC$_{S}$ performed slightly better than DE-MC in this example, and DE-MC$_{Z}$ slightly worse than DE-MC$_{ZS}$ (data not shown). The acceptance rate for each sampler in these runs varied between 0.14 and 0.17.

WinBugs runs of 10,000 iterations (taking twice the time of a DE-MC$_{ZS}$ run) that converged, gave about 3–5 times higher RMSE than DE-MC$_{ZS}$ for the percentiles of *lKa* and *lCl* and similar RMSEs for the other parameters (data not shown). Such runs converged to the correct values in *ca.* 50% of the cases.

5 Discussion

Standard DE-MC as proposed by ter Braak (2006) requires at least $N = 2d$ chains to be run in parallel. This is fine for low-dimensional problems, *i.e.* when only a relatively small number of parameters needs to be estimated. However, N parallel chains typically take N times longer to converge

than a single chain. This makes standard DE-MC rather inefficient for high-dimensional problems ($d > 20$, say). In this paper, we show that DE-MC can work for d up to 50–100 with far fewer chains (*e.g.* $N = 3$) by exploiting information from past samples from the individual chains. This approach extends the practical applicability of DE-MC. In a nonlinear mixed effects model example, the approach outperformed a block-update sampler geared to the specific features of the model. We chose $N = 3$ as this still allows for an accurate assessment of when convergence has been achieved. Nevertheless, for the problems considered herein $N = 1$ and $N = 4$ gave similar results.

The advantage of DE-MC$_{Z(S)}$ with N_z parallel chains over an N-chain DE-MC ($N_z \ll N$) can be understood as follows. For the same number of burn-in draws (CPU time), the burn-in of each member chain of DE-MC$_{Z(S)}$ is N/N_z times longer than that of each member chain of DE-MC. The extended burn-in per chain improves convergence in high dimensional problems when the initial population (initial distribution) is far from the target. The proposal generation by sampling difference vectors from the past had negligible effect on CPU time.

We observed from trace plots for a 100-dimensional Gaussian target that, after convergence, an N-chain DE-MC and an N-chain DE-MC$_{Z(S)}$ need about as many generations as the one-chain optimal RWMN needs iterations to move from an independent point on the target to another. These generations require N times more function evaluations, respectively, than the iterations of RWMN, but generate, according to the theorem for DE-MC in the Appendix and the corresponding conjecture for DE-MC$_{Z(S)}$, simultaneously N independent draws as compared a single independent draw for RWMN, yielding about equal efficiencies for Gaussian targets. The reported efficiencies of DE-MC and DE-MC$_{Z(S)}$ for Gaussian targets are less than 100, largely because of their shorter burn-in per member chain.

The new method, DE-MC$_{ZS}$, has two more algorithmic variables than DE-MC (M_0 and K). Our simulation and real data examples suggest that $M_0 = 10d$ is a reasonable value for the initial size of $\mathbf{Z}$. As there are almost no costs involved it would be tempting to set M_0 even higher, but this would hamper the speed of initial adaptation. The lag parameter K should not be too large either, as this would also hamper adaptation. Moreover, a small value of K would result in excessive storage. In all our runs, we chose K equal to the thinning rate. This choice works well for a range of problems, and avoids excessive storage requirements.

The DE-MC variants presented herein are among the simplest adaptive Metropolis sampling methods, yet attain high efficiency with respect to the optimal Normal jump Metropolis algorithm (Table 1). DE-MC$_{ZS}$ differs in various ways from the adaptive Metropolis sampler (AM) of Haario et al. (2001). AM runs a single chain and uses normally distributed jumps, using a covariance matrix that is based on all past samples. The method may have difficulty to start up and also to define the proper scaling factor γ for non-normal distributions. The Delayed Rejection added to AM in DRAM (Haario et al. 2006) helps to overcome some of these difficulties, although this method exhibits difficulty converging to the appropriate limiting distribution in the presence of multimodality. DE-MC$_{ZS}$ is quite similar in spirit to AM in that it uses past samples for adaptation to the posterior; DE-MC uses differences of past samples directly, where AM uses them indirectly via the covariance matrix. This store of past samples makes the memory requirement of DE-MC much larger than that of AM, but the computing time shorter. AM requires updating the covariance matrix at each sample and decomposing it regularly in triangular form, whereas DE-MC directly draws two past samples to generate a jump proposal.

No direct performance comparison between DE-MC and (DR)AM was attempted in this paper. An indirect comparison is as follows. AR and DRAM use normally distributed proposals, whereas these are suboptimal for distributions with heavier tails. AM and DRAM can therefore never be more efficient than the optimal RWMN and can thus never attain the 2–27 fold efficiency increase compared to the optimal RWMN sampler as DE-MC$_{ZS}$ attains in the Student t_3 example summarized in Tables 1 and 2. In addition, DE-MC can efficiently sample multimodal distributions (Strens et al. 2002; ter Braak 2006), whereas (DR)AM cannot.

The standard snooker update (Gilks et al. 1994; Gilks and Roberts 1996; Liang and Wong 2001) often requires a Metropolis step along the chosen direction. The DE-MC snooker update performs this Metropolis step with an adaptive step size. The proposed update is related to the Type IIb geometric proposal of Strens et al. (2002). Their proposal lacks, however, a default step size and the details of the Metropolis ratio of (4). When used in a 10–90 combination with the standard parallel direction update of (1) of DE-MC, the snooker update increased efficiency for extreme percentiles of the heavy tailed t_3 distribution (Table 1).

The good performance of DE-MC and its variants on the heavy tailed t distributions can be understood by noting that the proposed jumps are (once the chain is stationary) effectively sampling from a distribution with tails which are nearly as heavy as those of the target and are therefore more efficient than the light tailed jumps in RWMN (Roberts 2003; Jarner and Roberts 2007). DE-MC$_{ZS}$ benefits more from this than DE-MC and DE-MC$_S$ (Tables 1 and 2), because it uses fewer chains and convergence in each chain is inherently slow for a heavy tailed target (Jarner and Roberts 2007).

In our algorithm for DE-MC$_{Z(S)}$, each generated new sample $\mathbf{x}_i$ does not change $\mathbf{Z}$ immediately. This is advantageous in a multi-processor environment as it allows simultaneous real-time updating of the N chains. The outcome of

each processor is the Kth iterate of $\mathbf{x}_i$, which should then be stored in the matrix $\mathbf{Z}$ at a location that is accessible by all processors for proposal generation. In such environments, DE-MC and DE-MC$_{Z(S)}$ should perhaps be compared with an RWMN sampler with N parallel but independent chains. As the burn-in time of DE-MC, DE-MC$_{Z(S)}$ and RWMN are then of the same order (for identical N), the efficiency of DE-MC and DE-MC$_{Z(S)}$ compared to RWMN are likely to be higher than those reported in this paper. We confirmed this by simulation for Gaussian targets (results not shown).

This paper shows that DE-MC$_{ZS}$ can be a simple and attractive adaptive Metropolis sampler for d up to 50–100. For higher dimensions, we need to resort to block updating with block size up to 20–50 and combine block DE-MC with other multi-dimensional block-updaters, particularly when available in closed form. Such a mixed approach is already implemented in OpenBugs (Thomas and O'Hara 2007).

Acknowledgements The authors thank a referee of an earlier version of this manuscript for insights that greatly improved the convergence properties of the method and Jean-Luc Jannink for simulating discussions. The second author is supported by a J. Robert Oppenheimer Fellowship from the LANL postdoctoral program.

Appendix

The proof in ter Braak (2006) that Differential Evolution Markov Chain (DE-MC) is a valid MCMC method, contains an error, as was kindly pointed out by a referee. Here we give a valid proof, that is similar to that for the pinball sampler (Mengersen and Robert 2003). In essence, DE-MC works because the N chains can together be considered as a single chain on state space S^N, which is updated using an N-component Metropolis-within-Gibbs algorithm.

Let $\tilde{\pi}(\mathbf{x}_1,\ldots,\mathbf{x}_N)$be the target probability density function (pdf). Recall that a Metropolis-within-Gibbs algorithm is constructed in such a way that the jumping kernel $K_i(.|\mathbf{x}_1,\ldots,\mathbf{x}_N)$ for the ith component $\mathbf{x}_i$ satisfies for each $i=1,\ldots,N$ the conditional detailed balance condition with respect to $\tilde{\pi}(.|\mathbf{x}_1,\ldots,\mathbf{x}_{i-1},\mathbf{x}_{i+1},\ldots,\mathbf{x}_N)$:

$$\begin{aligned}&\tilde{\pi}(\mathbf{x}_i^{(t)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_{i+1}^{(t)},\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_i(\mathbf{x}_i^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_i^{(t)},\mathbf{x}_{i+1}^{(t)},\ldots,\mathbf{x}_N^{(t)})\\&=\tilde{\pi}(\mathbf{x}_i^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_i(\mathbf{x}_i^{(t)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_i^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)}).\end{aligned}\tag{5}$$

If $(\mathbf{x}_1^{(t)},\ldots,\mathbf{x}_N^{(t)})\sim\tilde{\pi}(.)$, the joint pdf of $\mathbf{x}_1,\ldots,\mathbf{x}_N$ at iteration t, it then follows that$(\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_N^{(t+1)})\sim\tilde{\pi}(.)$ at iteration $t+1$, because using (5) and the fact that kernels integrate to unity (adapted after Mengersen and Robert 2003)

$$\begin{aligned}&(\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_N^{(t+1)})\\&\sim\int\tilde{\pi}(\mathbf{x}_1^{(t)},\ldots,\mathbf{x}_N^{(t)})K_1(\mathbf{x}_1^{(t+1)}|\mathbf{x}_1^{(t)},\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_2(\mathbf{x}_2^{(t+1)}|\mathbf{x}_1^{(t+1)},\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\ldots\\&\quad\times K_N(\mathbf{x}_N^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{N-1}^{(t+1)},\mathbf{x}_N^{(t)})d\mathbf{x}_1^{(t)}\ldots d\mathbf{x}_N^{(t)}\\&=\int\tilde{\pi}(\mathbf{x}_1^{(t)}|\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_1(\mathbf{x}_1^{(t+1)}|\mathbf{x}_1^{(t)},\ldots,\mathbf{x}_N^{(t)}d\mathbf{x}_1^{(t)}\tilde{\pi}(\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_2(\mathbf{x}_2^{(t+1)}|\mathbf{x}_1^{(t+1)},\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\ldots\\&\quad\times K_N(\mathbf{x}_N^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{N-1}^{(t+1)},\mathbf{x}_N^{(t)})d\mathbf{x}_2^{(t)}\ldots d\mathbf{x}_N^{(t)}\\&=\int\tilde{\pi}(\mathbf{x}_1^{(t+1)}|\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\tilde{\pi}(\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_2(\mathbf{x}_2^{(t+1)}|\mathbf{x}_1^{(t+1)},\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\ldots\\&\quad\times K_N(\mathbf{x}_N^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{N-1}^{(t+1)},\mathbf{x}_N^{(t)})d\mathbf{x}_2^{(t)}\ldots d\mathbf{x}_N^{(t)}\\&=\int\tilde{\pi}(\mathbf{x}_1^{(t+1)},\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_2(\mathbf{x}_2^{(t+1)}|\mathbf{x}_1^{(t+1)},\mathbf{x}_2^{(t)},\ldots,\mathbf{x}_N^{(t)})\ldots\\&\quad\times K_N(\mathbf{x}_N^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{N-1}^{(t+1)},\mathbf{x}_N^{(t)})d\mathbf{x}_2^{(t)}\ldots d\mathbf{x}_N^{(t)}=\\&\vdots\\&=\int\tilde{\pi}(\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_i^{(t)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_i(\mathbf{x}_i^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_i^{(t)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\ldots\\&\quad\times K_N(\mathbf{x}_N^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{N-1}^{(t+1)},\mathbf{x}_N^{(t)})d\mathbf{x}_i^{(t)}\ldots d\mathbf{x}_N^{(t)}\\&=\int\tilde{\pi}(\mathbf{x}_i^{(t)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\\&\quad\times\tilde{\pi}(\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_i(\mathbf{x}_i^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_i^{(t)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\ldots\\&\quad\times K_N(\mathbf{x}_N^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{N-1}^{(t+1)},\mathbf{x}_N^{(t)})d\mathbf{x}_i^{(t)}\ldots d\mathbf{x}_N^{(t)}\\&=\int\tilde{\pi}(\mathbf{x}_i^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\\&\quad\times\tilde{\pi}(\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\\&\quad\times K_{i+1}(\mathbf{x}_{i+1}^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_i^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\ldots\\&\quad\times K_N(\mathbf{x}_N^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{N-1}^{(t+1)},\mathbf{x}_N^{(t)})d\mathbf{x}_{i+1}^{(t)}\ldots d\mathbf{x}_N^{(t)}\\&=\int\tilde{\pi}(\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_i^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\end{aligned}$$

$$\times K_{i+1}(\mathbf{x}_{i+1}^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_i^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)})\ldots$$
$$\times K_N(\mathbf{x}_N^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{N-1}^{(t+1)},\mathbf{x}_N^{(t)})d\mathbf{x}_{i+1}^{(t)}\ldots d\mathbf{x}_N^{(t)}$$
$$\vdots$$
$$=\tilde{\pi}(\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_N^{(t+1)}).$$

This concludes that proof that $\tilde{\pi}(.)$is the stationary distribution of the N-component Metropolis-within-Gibbs algorithm. We are now in the position to proof

Theorem 1 *DE-MC yields a Markov chain that is ergodic with unique stationary distribution* $(\mathbf{x}_1,\ldots,\mathbf{x}_N)\sim\tilde{\pi}(\mathbf{x}_1,\ldots,\mathbf{x}_N)=\pi(\mathbf{x}_1)\times\cdots\times\pi(\mathbf{x}_N)$.

Proof Chains are updated sequentially and conditionally upon one another. The proof consists of three parts. In part (a) we establish detailed balance of the kernel of the ith chain conditionally on the states of the other chains. We then use the general result on Metropolis within Gibbs in part (b) to proof that the chain has stationary distribution pdf $\pi(.)^N$. Ergodicity is proven in part (c).

(a) In this part we show that the kernel $K_i(.|\mathbf{x}_1,\ldots,\mathbf{x}_N)$of updating the ith chain satisfies for each $i=1,\ldots,N$ the conditional detailed balance condition with respect to $\pi(.)$:

$$\pi(\mathbf{x}_i^{(t)})K_i(\mathbf{x}_i^{(t+1)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_i^{(t)},\mathbf{x}_{i+1}^{(t)},\ldots,\mathbf{x}_N^{(t)})$$
$$=\pi(\mathbf{x}_i^{(t+1)})K_i(\mathbf{x}_i^{(t)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_i^{(t+1)},\mathbf{x}_{i+1}^{(t)}\ldots,\mathbf{x}_N^{(t)}).$$

The update of the ith chain proceeds by random selection of a pair of two different other chains. The kernel of this update thus is a mixture of $\binom{N-1}{2}$ kernels. Such a mixture maintains detailed balance with respect to $\pi(.)$, if each of its components does (Robert and Casella 2004). Let one such component be based on the pair of chains j and k ($j\neq i, k\neq i$, $j\neq k$). This component indeed maintains detailed balance as, from (1),

$$\mathbf{x}_i=\mathbf{x}^*-\gamma(\mathbf{x}_j-\mathbf{x}_k)-\mathbf{e}=\mathbf{x}^*+\gamma(\mathbf{x}_k-\mathbf{x}_j)-\mathbf{e} \tag{6}$$

and noting that $\mathbf{x}_j$ is selected equally often afore $\mathbf{x}_k$ as is $\mathbf{x}_k$ afore $\mathbf{x}_j$, and that the distribution of $\mathbf{e}$ is symmetric. Thus, detailed balance with respect to $\pi(.)$ is achieved pointwise by accepting the proposal with probability $\min(1,r)$ where $r=\pi(\mathbf{x}^*)/\pi(\mathbf{x}_i)$. As the Jacobian of the transformation implied by (6) is 1 in absolute value, detailed balance also holds in terms of arbitrary measurable sets, as required for reversibility of the Markov chain (Waagepetersen and Sorensen 2001).

(b) As shown in (a) the kernel $K_i(.|\mathbf{x}_1,\ldots,\mathbf{x}_N)$of updating the ith chain satisfies for each $i=1,\ldots,N$ the conditional detailed balance condition (5) with

$$\tilde{\pi}(\mathbf{x}_i^{(t)}|\mathbf{x}_1^{(t+1)},\ldots,\mathbf{x}_{i-1}^{(t+1)},\mathbf{x}_{i+1}^{(t)},\ldots,\mathbf{x}_N^{(t)})=\pi(\mathbf{x}_i^{(t)}).$$

DE-MC is thus an N-component Metropolis-within-Gibbs algorithm with joint stationary distribution$(\mathbf{x}_1,\ldots,\mathbf{x}_N)\sim\tilde{\pi}(\mathbf{x}_1,\ldots,\mathbf{x}_N)=\pi(\mathbf{x}_1)\times\cdots\times\pi(\mathbf{x}_N)$.

(c) The stationary distribution is unique, if the chain is aperiodic, not transient and irreducible (Robert and Casella 2004). The first two conditions are satisfied, except for trivial exceptions, because of the random walk component generated by $\mathbf{e}$ in each DE-MC update. For the third condition, it is required that any state can be reached with positive probability and this is guaranteed by the unbounded support of the distribution of $\mathbf{e}$ in (6) (Robert and Casella 2004). Each component has therefore a unique stationary distribution which, from (a), is $\pi(.)$. This concludes the proof. □

References

Babu, B.V., Angira, R.: Modified differential evolution (MDE) for optimization of non-linear chemical processes. Comput. Chem. Eng. **30**, 989–1002 (2006)

Gelman, A., Carlin, J.B., Stern, H.S., Rubin, D.B.: Bayesian Data Analysis, 2nd edn. Chapman & Hall, London (2004)

Gilks, W.R., Roberts, G.O.: Strategies for improving MCMC. In: Gilks, W.R., Richardson, S., Spiegelhalter, D.J. (eds.) Markov Chain Monte Carlo in Practice, pp. 89–114. Chapman & Hall, London (1996)

Gilks, W.R., Roberts, G.O., George, E.I.: Adaptive direction sampling. Statistician **43**, 179–189 (1994)

Haario, H., Saksman, E., Tamminen, J.: An adaptive Metropolis algorithm. Bernoulli **7**, 223–242 (2001)

Haario, H., Laine, M., Mira, A., Saksman, E.: DRAM: Efficient adaptive MCMC. Stat. Comput. **16**, 339–354 (2006)

Jarner, S.F., Roberts, G.O.: Convergence of heavy-tailed Monte Carlo Markov chain algorithms. Scand. J. Stat. **34**, 781–815 (2007)

Liang, F.M., Wong, W.H.: Real-parameter evolutionary Monte Carlo with applications to Bayesian mixture models. J. Am. Stat. Assoc. **96**, 653–666 (2001)

Liu, J., Hodges, J.S.: Posterior bimodality in the balanced one-way random-effects model. J. R. Stat. Soc. Ser. B **65**, 247–255 (2003)

Mengersen, K., Robert, C.P.: IID sampling using self-avoiding population Monte Carlo: the pinball sampler. In: Bernardo, J.M., Bayarri, M.J., Berger, J.O., Dawid, A.P., Heckerman, D., Smith, A.F.M., West, M. (eds.) Bayesian Statistics 7, pp. 277–292. Clarendon, Oxford (2003)

Pinheiro, J.C., Bates, D.M.: Mixed-Effects Models in S and S-PLUS. Springer, New York (2000)

Price, K.V., Storn, R.M., Lampinen, J.A.: Differential Evolution, A Practical Approach to Global Optimization. Springer, Berlin (2005)

Robert, C.P., Casella, G.: Monte Carlo Statistical Methods, 2nd edn. Springer, New York (2004)

Roberts, G.O., Gilks, W.R.: Convergence of adaptive direction sampling. J. Multivar. Anal. **49**, 287–298 (1994)

Roberts, G.O., Rosenthal, J.S.: Optimal scaling for various Metropolis-Hastings algorithms. Stat. Sci. **16**, 351–367 (2001)

Roberts, G.O.: Linking theory and practice of MCMC. In: Green, P.J., Hjort, N.L., Richardson, S. (eds.) Highly Structured Stochastic Systems, pp. 145–166. Oxford University Press, Oxford (2003)

Roberts, G.O., Rosenthal, J.S.: Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. J. Appl. Probab. **44**, 458–475 (2007)

Roberts, G.O., Rosenthal, J.S.: Examples of adaptive MCMC. Online ms. (2008)

Spiegelhalter, D., Thomas, A., Best, N., Lunn, D.: WinBUGS User Manual version 1.4. www.mrc-bsu.cam.ac.uk/bugs (2003)

Storn, R., Price, K.: Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. **11**, 341–359 (1997)

Strens, M., Bernhardt, M., Everett, N.: Markov chain Monte Carlo sampling using direct search optimization. In: Sammut, C., Hoffmann, A.G. (eds.) Machine Learning, Proceedings of the Nineteenth International Conference on Machine Learning (ICML 2002), pp. 602–609. Morgan Kaufmann, San Fransisco (2002)

ter Braak, C.J.F.: A Markov chain Monte Carlo version of the genetic algorithm differential evolution: easy Bayesian computing for real parameter spaces. Stat. Comput. **16**, 239–249 (2006)

Thomas, A., O'Hara, R.B.: OpenBUGS. http://mathstat.helsinki.fi/openbugs/ (2007)

Vrugt, J.A., ter Braak, C.J.F., Gupta, H.V., Robinson, B.A.: Equifinality of formal (DREAM) and informal (GLUE) Bayesian approaches in hydrologic modeling? Stochastic Environmental Research and Risk Assessment (SERRA) (2008a). DOI:10.1007/s00477-008-0274-y

Vrugt, J.A., ter Braak, C.J.F., Clark, M.P., Hyman, J.M., Robinson, B.A.: Treatment of input uncertainty in hydrologic modeling: doing hydrology backwards with Markov chain Monte Carlo simulation. Water Resour. Res. (2008b, in press)

Waagepetersen, R., Sorensen, D.: A tutorial on reversible jump MCMC with a view toward applications in QTL-mapping. Int. Stat. Rev. **69**, 49–61 (2001)